T291 Management Information System

MEETING 6-7: THE RELATIONAL DATA MODEL

AOU UNIVERSITY

Relational Model Concepts

A Relation is a mathematical concept based on the ideas of sets

The model was first proposed by Dr. E.F. Codd of IBM Research in 1970 in the following paper:

- "A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970

The above paper caused a major revolution in the field of database management and earned Dr. Codd the coveted ACM Turing Award

and Relational Database Constraints

Most widely used model.

- Vendors: IBM, Informix, Microsoft, Oracle, Sybase, etc.
- First commercial implementations available in early 1980s
- Has been implemented in a large number of commercial system

Hierarchical and network models

- Preceded the relational model

“Legacy systems” in older models

- E.G., IBM’s IMS

Recent competitor: object-oriented model

- ObjectStore, Versant, Ontos
- A synthesis emerging: *object-relational model*
  - Informix Universal Server, UniSQL, O2, Oracle, DB2

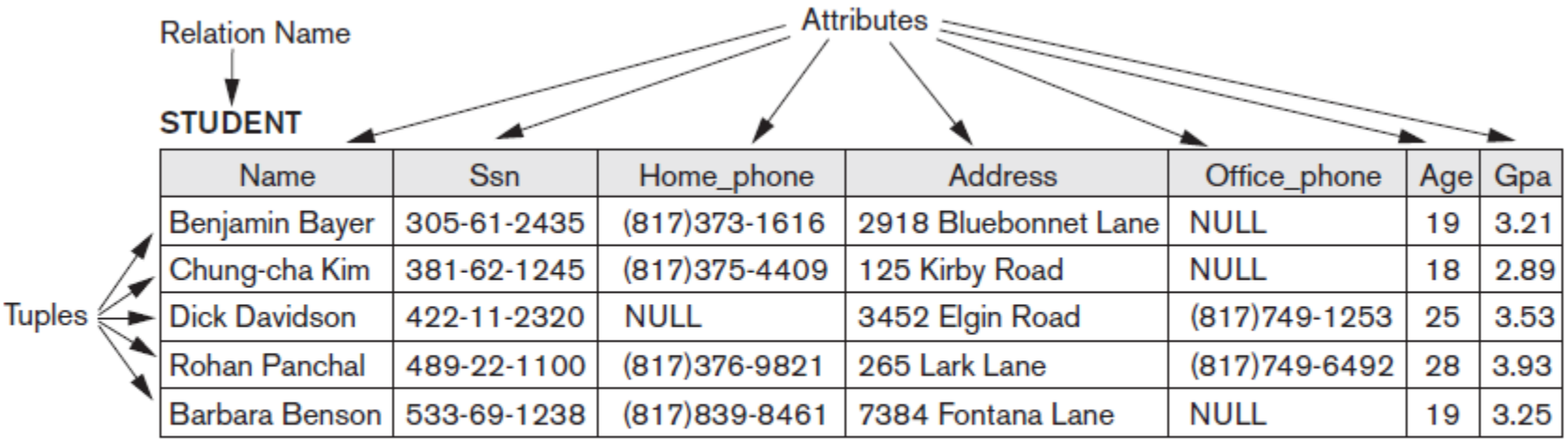
Relational Model Concepts

Represents data as a collection of relations

Table of values

- Row
  - Represents a collection of related data values
  - Fact that typically corresponds to a real-world entity or relationship
  - *Tuple*
- Table name and column names
  - Interpret the meaning of the values in each row *attribute*

Relational Model Concepts (cont'd.)



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 3.1
The attributes and tuples of a relation STUDENT.

Domains, Attributes, Tuples, and Relations

Domain D

- Set of atomic values

Atomic

- Each value indivisible

Specifying a domain

- **Data type** specified for each domain

Formal Definitions - Domain

A **domain** has a logical definition:

- Example: “USA_phone_numbers” are the set of 10 digit phone numbers valid in the U.S.

A domain also has a data-type or a format defined for it.

- The USA_phone_numbers may have a format: (ddd)ddd-dddd where each d is a decimal digit.
- Dates have various formats such as year, month, date formatted as yyyy-mm-dd, or as dd mm,yyyy etc.

The attribute name designates the role played by a domain in a relation:

- Used to interpret the meaning of the data elements corresponding to that attribute
- Example: The domain Date may be used to define two attributes named “Invoice-date” and “Payment-date” with different meanings

Domains, Attributes, Tuples, and Relations (cont'd.)

Relation schema R

- Denoted by $R(A_1, A_2, ..., A_n)$
- Made up of a relation name R and a list of attributes, $A_1, A_2, ..., A_n$

Attribute A_i

- Name of a role played by some domain D in the relation schema R

Degree (or **arity**) of a relation

- Number of attributes n of its relation schema
- #fields = *degree / arity*

Domains, Attributes, Tuples, and Relations (cont'd.)

Relation (or **relation state**)

- Set of ***n*-tuples** $r = \{t_1, t_2, ..., t_m\}$
- Each n-tuple t
  - Ordered list of n values $t = <v_1, v_2, ..., v_n>$
  - Each value v_i, $1 \leq i \leq n$, is an element of dom(A_i) or is a special NULL value

Domains, Attributes, Tuples, and Relations (cont'd.)

Relation (or relation state) $r(R)$

- **Mathematical relation** of degree n on the domains $\text{dom}(A_1)$, $\text{dom}(A_2)$, ..., $\text{dom}(A_n)$
- **Subset** of the **Cartesian product** of the domains that define R:
  - $r(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times ... \times \text{dom}(A_n))$

Domains, Attributes, Tuples, and Relations (cont'd.)

Cardinality

- Total number of values in domain
- #Rows = *cardinality*

Current relation state

- Relation state at a given time
- Reflects only the valid tuples that represent a particular state of the real world

Attribute names

- Indicate different **roles**, or interpretations, for the domain

Characteristics of Relations

Ordering of tuples in a relation

- Relation defined as a set of tuples
- Elements have no order among them

Ordering of values within a tuple and an alternative definition of a relation

- Order of attributes and values is not that important
- As long as correspondence between attributes and values maintained

Characteristics of Relations (cont'd.)

Alternative definition of a relation

- Tuple considered as a set of (<attribute>, <value>) pairs
- Each pair gives the value of the mapping from an attribute A_i to a value v_i from dom(A_i)

Use the first definition of relation

- Attributes and the values within tuples are ordered
- Simpler notation

Characteristics of Relations (cont'd.)

Figure 3.2
The relation STUDENT from Figure 3.1 with a different order of tuples.

STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21

Characteristics of Relations (cont'd.)

Values and NULLs in tuples

- Each value in a tuple is atomic
- **Flat relational model**
  - Composite and multivalued attributes not allowed
  - **First normal form** assumption
- Multivalued attributes
  - Must be represented by separate relations
- Composite attributes
  - Represented only by simple component attributes in basic relational model

Characteristics of Relations (cont'd.)

NULL values

- Represent the values of attributes that may be unknown or may not apply to a tuple
- Meanings for NULL values
  - *Value unknown*
  - *Value exists but is not available*
  - *Attribute does not apply to this tuple (also known as value undefined)*

Characteristics of Relations (cont'd.)

Interpretation (meaning) of a relation

- **Assertion**
  - Each tuple in the relation is a **fact** or a particular instance of the assertion
- **Predicate**
  - Values in each tuple interpreted as values that satisfy predicate

Relational Model Notation

Relation schema R of degree n

- Denoted by $R(A_1, A_2, ..., A_n)$

Uppercase letters Q, R, S

- Denote relation names

Lowercase letters q, r, s

- Denote relation states

Letters t, u, v

- Denote tuples

Relational Model Notation

Name of a relation schema: STUDENT

- Indicates the current set of tuples in that relation

Notation: STUDENT(Name, Ssn, ...)

- Refers only to relation schema

Attribute *A* can be qualified with the relation name *R* to which it belongs

- Using the dot notation *R.A*

Relational Model Notation

n-tuple t in a relation $r(R)$

- Denoted by $t = <v_1, v_2, ..., v_n>$
- v_i is the value corresponding to attribute A_i

Component values of tuples:

- $t[A_i]$ and $t.A_i$ refer to the value v_i in t for attribute A_i
- $t[A_u, A_w, ..., A_z]$ and $t.(A_u, A_w, ..., A_z)$ refer to the subtuple of values $<v_u, v_w, ..., v_z>$ from t corresponding to the attributes specified in the list

Relational Model Constraints

Constraints

- Restrictions on the actual values in a database state
- Derived from the rules in the miniworld that the database represents

Inherent model-based constraints or **implicit constraints**

- Inherent in the data model

Relational Model Constraints (cont'd.)

Schema-based constraints or **explicit constraints**

- Can be directly expressed in schemas of the data model

Application-based or **semantic constraints** or **business rules**

- Cannot be directly expressed in schemas
- Expressed and enforced by application program

Formal Definitions - Summary

Formally,

- Given R(A1, A2, ……., An)
- $r(R) \subset dom(A1) \times dom(A2) \times \ldots \times dom(An)$

R(A1, A2, …, An) is the **schema** of the relation

R is the **name** of the relation

A1, A2, …, An are the **attributes** of the relation

r(R): a specific **state** (or "value" or “population”) of relation R – this is a *set of tuples* (rows)

- r(R) = {t1, t2, …, tn} where each ti is an n-tuple
- ti = <v1, v2, …, vn> where each vj *element-of* dom(Aj)

Formal Definitions - Example

Let R(A1, A2) be a relation schema:

- Let dom(A1) = {0,1}
- Let dom(A2) = {a,b,c}

Then: dom(A1) X dom(A2) is all possible combinations:

{<0,a> , <0,b> , <0,c>, <1,a>, <1,b>, <1,c> }

The relation state r(R) $\subset$ dom(A1) X dom(A2)

For example: r(R) could be {<0,a> , <0,b> , <1,c> }

- this is one possible state (or "population" or "extension") r of the relation R, defined over A1 and A2.

Definition Summary

Informal Terms	Formal Terms
Table	Relation
Column Header	Attribute
All possible Column Values	Domain
Row	Tuple
Table Definition	Schema of a Relation
Populated Table	State of the Relation

Domain Constraints

Typically include:

- Numeric data types for integers and real numbers
- Characters
- Booleans
- Fixed-length strings
- Variable-length strings
- Date, time, timestamp
- Money
- Other special data types

Key Constraints and Constraints on NULL Values

No two tuples can have the same combination of values for all their attributes.

Superkey

- No two distinct tuples in any state *r* of *R* can have the same value for SK

Key

- Superkey of R
- Removing any attribute *A* from *K* leaves a set of attributes *K* that is not a superkey of *R* any more

Key Constraints and Constraints on NULL Values (cont'd.)

Key satisfies two properties:

- Two distinct tuples in any state of relation cannot have identical values for (all) attributes in key
- Minimal superkey
  - Cannot remove any attributes and still have uniqueness constraint in above condition hold

Key Constraints and Constraints on NULL Values (cont'd.)

Candidate key

- Relation schema may have more than one key

Primary key of the relation

- Designated among candidate keys
- Underline attribute

Other candidate keys are designated as **unique keys**

Key Constraints and Constraints on NULL Values (cont'd.)

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

Figure 3.4
The CAR relation, with two candidate keys: License_number and Engine_serial_number.

Relational Databases and Relational Database Schemas

Relational database schema S

- Set of relation schemas $S = \{R_1, R_2, ..., R_m\}$
- Set of integrity constraints IC

Relational database state

- Set of relation states $DB = \{r_1, r_2, ..., r_m\}$
- Each r_i is a state of R_i and such that the r_i relation states satisfy integrity constraints specified in IC

Integrity, Referential Integrity, and Foreign Keys

Entity integrity constraint

- No primary key value can be NULL

Referential integrity constraint

- Specified between two relations
- Maintains consistency among tuples in two relations

Integrity, and Foreign Keys (cont'd.)

Foreign key rules:

- The attributes in FK have the same domain(s) as the primary key attributes PK
- Value of FK in a tuple t_1 of the current state $r_1(R_1)$ either occurs as a value of PK for some tuple t_2 in the current state $r_2(R_2)$ or is NULL

Integrity, and Foreign Keys (cont’d.)

Diagrammatically display referential integrity constraints

- Directed arc from each foreign key to the relation it references

All integrity constraints should be specified on relational database schema

Other Types of Constraints

Semantic integrity constraints

- May have to be specified and enforced on a relational database
- Use **triggers** and **assertions**
- More common to check for these types of constraints within the application programs

Other Types of Constraints (cont'd.)

Functional dependency constraint

- Establishes a functional relationship among two sets of attributes *X* and *Y*
- Value of *X* determines a unique value of *Y*

State constraints

- Define the constraints that a valid state of the database must satisfy

Transition constraints

- Define to deal with state changes in the database

Update Operations, Transactions, and Dealing with Constraint Violations

Operations of the relational model can be categorized into retrievals and updates

Basic operations that change the states of relations in the database:

- Insert
- Delete
- Update (or Modify)

Figure 3.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Figure 3.6
One possible database state for the COMPANY relational database schema.

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

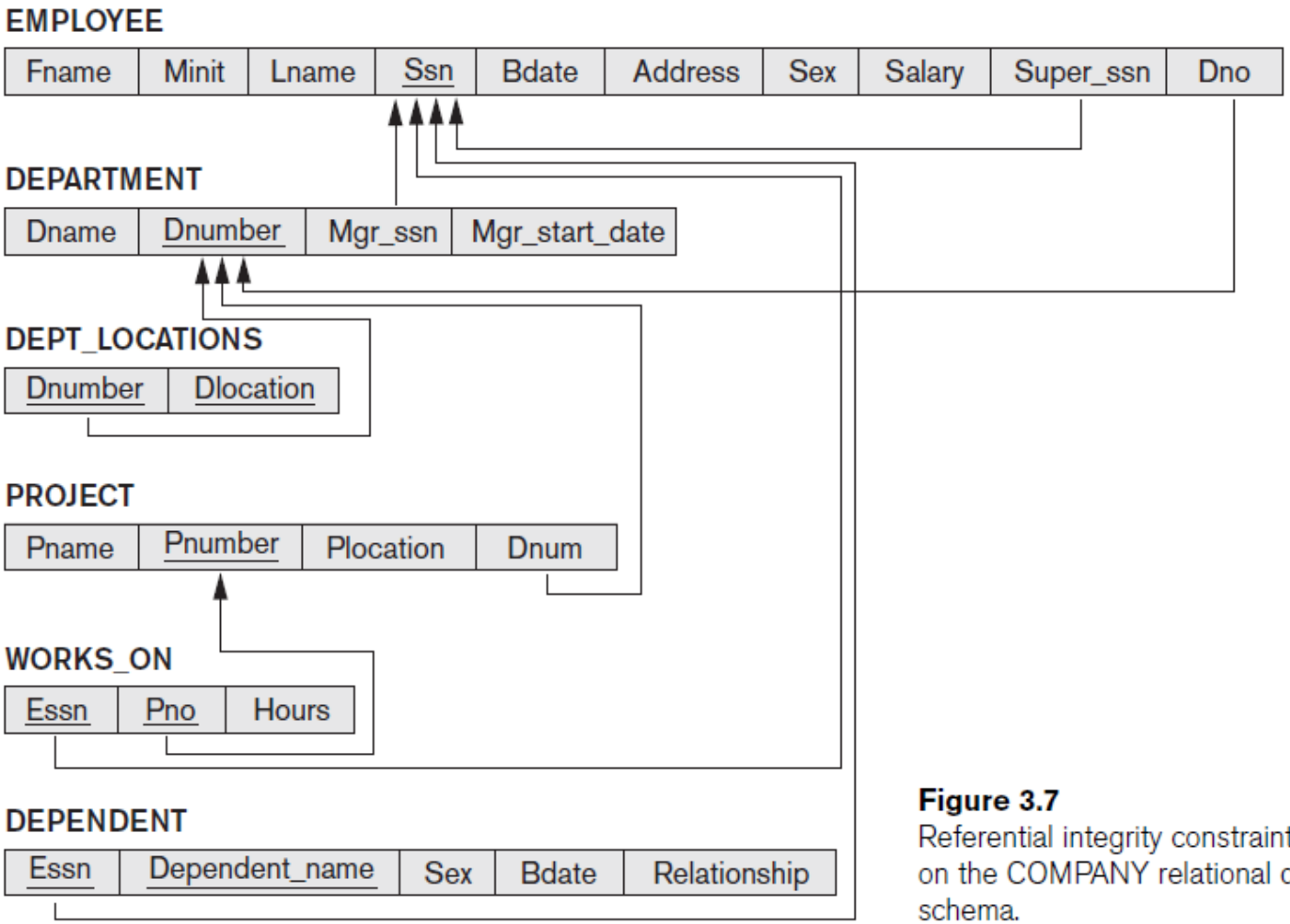


Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

The Insert Operation

Provides a list of attribute values for a new tuple

t that is to be inserted into a relation R

Can violate any of the four types of constraints

If an insertion violates one or more constraints

- Default option is to reject the insertion

The Delete Operation

Can violate only referential integrity

- If tuple being deleted is referenced by foreign keys from other tuples
- **Restrict**
  - Reject the deletion
- **Cascade**
  - Propagate the deletion by deleting tuples that reference the tuple that is being deleted
- **Set null** or **set default**
  - Modify the referencing attribute values that cause the violation

The Update Operation

Necessary to specify a condition on attributes of relation

- Select the tuple (or tuples) to be modified

If attribute not part of a primary key nor of a foreign key

- Usually causes no problems

Updating a primary/foreign key

- Similar issues as with Insert/Delete

The Transaction Concept

Transaction

- Executing program
- Includes some database operations
- Must leave the database in a valid or consistent state

Online transaction processing (OLTP) systems

- Execute transactions at rates that reach several hundred per second

Summary

Characteristics differentiate relations from ordinary tables or files

Classify database constraints into:

- Inherent model-based constraints, explicit schema-based constraints, and application-based constraints

Modification operations on the relational model:

- Insert, Delete, and Update